$\mathsf{QMIP} = \mathsf{MIP}^*$

Anne Broadbent[1], Joseph Fitzsimons[1,2], Elham Kashefi [3] *

Abstract

The way entanglement influences the power of quantum and classical multi-prover interactive proof systems is a long-standing open question. We show that the class of languages recognized by quantum multi-prover interactive proof systems, QMIP, is equal to MIP^*, the class of languages recognized by classical multi-prover interactive proof systems where the provers share entanglement. After the recent result by Jain, Ji, Upadhyay and Watrous showing that $\mathsf{QIP} = \mathsf{IP}$, our work completes the picture from the verifier's perspective by showing that also in the setting of multiple provers with shared entanglement, a quantum verifier is no more powerful than a classical one: $\mathsf{QMIP} = \mathsf{MIP}^*$. Our techniques are based on the adaptation of universal blind quantum computation (a protocol recently introduced by us) to the context of interactive proof systems. We show that in the multi-prover scenario, shared entanglement has a positive effect in removing the need for a quantum verifier. As a consequence, our results show that the entire power of quantum information in multi-prover interactive proof systems is captured by the shared entanglement and not by the quantum communication.

[1]Institute for Quantum Computing, University of Waterloo, Waterloo, Ontario, Canada.
[2]Department of Materials, University of Oxford, Oxford, United Kingdom.
[3]School of Informatics, University of Edinburgh, Edinburgh, Scotland, United Kingdom.
*Email: albroadb@iqc.ca, joe.fitzsimons@materials.ox.ac.uk, ekashefi@inf.ed.ac.uk

1 Introduction and Related work

An interactive proof system [GMR89, Bab85] consists of an interaction between a computationally unbounded prover and a computationally bounded probabilistic verifier. The prover attempts to convince the verifier that a given input string satisfies some property, while the verifier tries to determine the validity of this proof. A language L is said to have an interactive proof system if there exists a randomized polynomial-time verifier V such that an honest prover can, with high probability, convince V to accept when the given input is in L (completeness), and no prover can convince V to accept with high probability when the input is not in L (soundness). The class of languages having interactive proof systems is denoted IP. Multi-prover interactive proofs (MIP), first proposed in [BOGKW88], are the generalization of interactive proofs to the multi-prover scenario. It was shown in [BFL91, FRS94] that $\mathsf{MIP} = \mathsf{NEXP}$, paving the way to important results in inapproximability and probabilistically checkable proofs [FGL$^+$96, AS98, ALM$^+$98].

The quantum analogue of interactive proofs, quantum interactive proofs (QIP), were first introduced by Watrous [Wat99]: they involve a computationally unbounded prover exchanging quantum messages with a polynomially bounded quantum verifier. Using the powerful techniques of capturing the computational power of quantum interactive proofs by semi-definite programming (SDP), Kitaev and Watrous showed [KW00] that $\mathsf{IP} \subseteq \mathsf{QIP} = \mathsf{QIP}(3) \subseteq \mathsf{EXP}$ (where $\mathsf{QIP}(k)$ denotes a k-message quantum interactive proof). Recently, by employing an efficient parallel algorithm for SDP, Jain, Ji, Upadhyay and Watrous [JJUW09] solved a long-standing open problem on the power of quantum interactive proof systems by showing that QIP is contained in PSPACE (since $\mathsf{IP} = \mathsf{PSPACE}$ [LFKN90, Sha90] it follows that $\mathsf{QIP} = \mathsf{IP}$); this work builds on a previous result that $\mathsf{QIP}(2) \subseteq \mathsf{PSPACE}$ [JUW09]. We therefore conclude that quantum information adds no power to the single-prover interactive proof scenario. The corresponding question regarding the power of multi-prover quantum interactive proof systems where the provers share prior entanglement but otherwise cannot communicate, however, remains open.

Quantum interactive proofs with multiple provers (QMIP) were introduced by Kobayashi and Matsumoto [KM03], where they proved that in the case where provers share no entanglement, $\mathsf{QMIP}_{(\text{n.e.})} = \mathsf{MIP}$ and moreover when the provers share at most polynomially many entangled qubits, $\mathsf{QMIP}_{(\text{l.e.})} \subseteq \mathsf{NEXP}$. Several papers have already analyzed both negative and positive aspects of sharing entanglement in the context of interactive proofs with multiple provers involving a quantum verifier (QMIP) or a classical verifier (MIP^*) [CHTW04, KKMV08, KV06] and yet the question of how entanglement influences the power of such proof systems has not been answered: since entanglement can potentially increase both the completeness and soundness error, it is not even clear whether the expressive power of either QMIP or MIP^* is a subset or superset of, or is incomparable to NEXP.

Hence one could hope for a breakthrough using fresh techniques for a full understanding of the expressive power of quantum multi-prover interactive proofs with provers sharing an unlimited amount of entanglement. This paper presents a step forward in this direction: based on a novel approach connecting a cryptographic protocol with interactive proof systems, we show that a quantum verifier is no more powerful than a classical verifier even in the multi-party scenario, and hence that $\mathsf{QMIP} = \mathsf{MIP}^*$.

2 Summary of Contributions and Techniques

Recently Jain, Ji, Upadhyay and Watrous [JJUW09] obtained the surprising result that quantum interactive proof systems and classical interactive proof systems have equivalent expressive power, proving that $\mathsf{QIP} = \mathsf{IP}$. Using a different approach based on a cryptographic protocol, we prove the analogue of this result in the context of multiple provers with shared entanglement, to demonstrate that quantum computing adds no power to interactive proof systems even with multiple provers: $\mathsf{QMIP} = \mathsf{MIP}^*$. More precisely, we show that for any number k of provers, $\mathsf{QMIP}[k] = \mathsf{MIP}^*[k]$ (the case $k = 1$ following from [JJUW09]). Our proof is heavily based on the usage of shared entanglement which can be seen as another positive aspect of shared entanglement in the multi-prover scenario.

Our techniques are based on the adaptation of universal blind quantum computation [BFK09]

(a protocol recently introduced by us) to the context of interactive proof systems. We have already used this method to show that any language in BQP has an entangled two-prover interactive proof system requiring a verifier performing only randomized polynomial-time classical computation and provers performing polynomial-time quantum computation. Similarly, we show in this work how to use the shared entanglement between computationally unbounded provers to reduce the requirement of the verifier in a multi-prover quantum interactive proof system to purely classical computation, without reducing the expressive power of the interactive proof system. The connection between interactive proof system and blind quantum computation was first studied by Aharonov, Ben-Or and Eban [ABE10], however it remains an open question whether their setting can yield a multi-prover interactive proof system with a purely classical verifier, which is what we require for our construction.

Informally speaking, a protocol $\mathcal{P}$ for a language L in QMIP consists of several rounds where the verifier receives a quantum message from each prover which she processes to generate the next set of quantum messages to be sent to the provers. One can use the blind quantum computation protocol between the verifier and entangled provers to remove this need for the verifier to exchange and process quantum information with the provers. In order to run a quantum circuit, she enlists the help of two entangled provers, P_1 and P_2. P_1 first receives authenticated and encrypted quantum messages teleported [BBC$^+$93] from the other provers (the special case of authenticating P_1's own message is also covered), and then the two-server blind quantum computation protocol is executed with P_1 and P_2, so that P_1 eventually computes the next set of messages. These messages are redistributed (via teleportation) to all other provers, with the help of whom the verifier then verifies the authenticity of the messages. Note that we use a loose definition of teleportation, referring to the special case where the verifier controls the classical communication between provers since in our protocol, there is no direct communication between provers. Moreover, using encryption and an authentication code, the verifier can do away with her need to store a local quantum register between rounds by using P_1's memory. The blindness property guarantees that no information is leaked to the provers, as though the verifier herself is running the quantum circuits. Repeating this process, we simulate $\mathcal{P}$ with a protocol requiring only a classical verifier, thus showing that $L \in \mathsf{MIP}^*$. The overhead cost of such action is the usage of a linear (in the size of the verifier's quantum circuits) number of copies of entangled states $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ and polynomial classical communication. We prove that the completeness is unchanged and that the soundness error can be bounded arbitrary close to the original value.

3 Preliminaries

We assume that the reader is familiar with the quantum formalism, including the quantum circuit model and measurement-based quantum computing (MBQC) [RB01, RBB03, DKP07]. This section reviews the model of quantum multi-prover interactive proof systems and sketches the blind quantum computation protocol.

We use the definition and notation defined in the earlier work of Kempe, Kobayashi, Matsumoto and Vidick [KKMV08]. A quantum k-prover interactive proof system consists of a verifier V with private quantum register V (with one qubit designated as the *output qubit*) and k provers $P_1, \ldots, P_k$ with private quantum registers $\mathsf{P_1}, \ldots, \mathsf{P_k}$, as well as quantum message registers $\mathsf{M_1}, \ldots, \mathsf{M_k}$. At the beginning of the protocol, all the qubits in $(\mathsf{V}, \mathsf{M_1}, \ldots, \mathsf{M_k})$ are initialized to $|0 \ldots 0\rangle$, and the qubits in $(\mathsf{P_1}, \ldots, \mathsf{P_k})$ are in some shared state, $|\Phi\rangle$, prepared by the provers in advance (and hence possibly entangled). No communication between the provers is allowed after the preparation of this state. The protocol consists of alternating turns of the provers and of the verifier, starting with the verifier, if m is even, and with the provers otherwise. During the turn of the verifier, V applies some polynomial-time circuit to the qubits in $(\mathsf{V}, \mathsf{M_1}, \ldots, \mathsf{M_k})$, and then sends each message register $\mathsf{M_i}$ to prover P_i. During the turn of the provers, each P_i applies some transformation to the registers $(\mathsf{P_i}, \mathsf{M_i})$ for $1 \leq i \leq k$ and sends $\mathsf{M_i}$ back to the verifier. The last turn is always a turn for the provers. After the last turn, the verifier applies a polynomial-time circuit to the qubits in $(\mathsf{V}, \mathsf{M_1}, \ldots, \mathsf{M_k})$, and then measures the output qubit in the standard basis, accepting if the outcome is $|1\rangle$ and rejecting otherwise.

Formally, an *m-turn polynomial-time quantum verifier* V for k-prover QMIP system is a polynomial-time computable mapping from input strings x to a set of polynomial-time uniformly generated circuits $\{V^1, \ldots, V^{\lceil (m+1)/2 \rceil}\}$, and a partition of the space on which they act into registers $(\mathsf{V}, \mathsf{M_1}, \ldots, \mathsf{M_k})$, which consist of polynomially many qubits. Similarly, an *m-turn quantum prover* P is a mapping from x to a set of circuits $\{P^1, \ldots, P^{\lceil (m+1)/2 \rceil}\}$ each acting on registers $(\mathsf{P_i}, \mathsf{M_i})$. No restrictions are placed on the complexity of this mapping or the size of $\mathsf{P_i}$, however, the provers are limited by the linearity of quantum mechanics. Note that in our setting, circuits V^i and P^i might include measurements (e.g. a measurement pattern) and implement a general completely positive trace preserving map, and not just a unitary operator, however this is equivalent to the standard setting [AKN98, RBB03].

The *protocol* $(V, P_1, \ldots, P_k, |\Phi\rangle)$ is the alternating application of provers' and verifier's circuits to the state $|0 \ldots 0\rangle \otimes |\Phi\rangle$ in registers $(\mathsf{V}, \mathsf{M_1}, \ldots, \mathsf{M_k}, \mathsf{P_1}, \ldots, \mathsf{P_k})$. We say that $(V, P_1, \ldots, P_k, |\Phi\rangle)$ accepts x if the designated output qubit in V is measured in $|1\rangle$ at the end of the protocol and call the probability with which this happens $p_{acc}(x, V, P_1, \ldots, P_k, |\Phi\rangle)$.

Definition 1. *A language L is in $\mathsf{QMIP}(k, m, c, s)$ if there exists an m-turn polynomial-time quantum verifier V for quantum k-prover interactive proof systems such that, for every input x:*

(Completeness) *if $x \in L$, there exist m-turn quantum provers $P_1, \ldots, P_k$ and a shared state $|\Phi\rangle$ such that $p_{acc}(x, V, P_1, \ldots, P_k, |\Phi\rangle) \geq c$,*

(Soundness) *if $x \notin L$, for any m-turn quantum provers $P'_1, \ldots, P'_k$ and any shared state $|\Phi'\rangle$, $p_{acc}(x, V, P'_1, \ldots, P'_k, |\Phi'\rangle) \leq s$. (We refer to s as the* soundness error.*)*

We can similarly define an entangled k-prover interactive proof system (MIP^*) to be a quantum k-prover interactive proof system where the verifier's private and message registers are classical, and with the unbounded provers performing quantum computations and having access to shared entanglement. Hence an *m-turn polynomial-time classical verifier* V for a k-prover MIP^* system is a polynomial-time computable mapping from input strings x to a set of polynomial-time uniformly generated classical circuits $\{V^1, \ldots, V^{\lceil (m+1)/2 \rceil}\}$, and a partition of the space on which they act into registers $(\mathsf{V}, \mathsf{M_1}, \ldots, \mathsf{M_k})$, which consist of polynomially many classical bits. An *m-turn quantum prover* P is a mapping from x to a set of circuits $\{P^1, \ldots, P^{\lceil (m+1)/2 \rceil}\}$, each acting on registers $(\mathsf{M_i}, \mathsf{P_i})$. Circuits V^i and P^i might employ private randomness. At the beginning of the protocol, all the bits in $(\mathsf{M_1}, \ldots, \mathsf{M_k})$ are initialized to zero, and the qubits in $(\mathsf{P_1}, \ldots, \mathsf{P_k})$ are in some shared state, $|\Phi\rangle$, prepared by the provers in advance (and hence possibly entangled). The remaining definition are exactly the same as for QMIP.

Next we briefly discuss the universal blind quantum computation protocol (UBQC) [BFK09] which will be used in the proof of our main result. Suppose Alice has in mind a unitary operator U that is implemented with a pattern on a fixed but universal graph state with (X, Y) measurements with angles given as multiples of $\pi/4$. This pattern could have been designed either directly in MBQC or from a circuit construction. Alice does not have the full quantum power to implement U, and wishes to use Bob as a resource to do so, while maintaining the privacy of her computation, meaning that Bob does not learn anything about the computation that he is helping Alice perform (except an upper bound on the dimensions of her circuit). Following [BFK09], we say that such a protocol is *blind* if Bob's view of the protocol does not depend on Alice's input (X), when given an upper bound on the dimensions of her circuit (Y); since his view consists of classical and quantum information, this means that the distribution of the classical information does not depend on X (given Y) and that for any fixed choice of the classical information, the state of the quantum system is uniquely determined and does not depend on X (given Y). We will subsequently refer to this property as *blindness*.

There are two stages to the protocol: *preparation* and *computation*. In the preparation stage, Alice prepares single qubits chosen randomly from $\{\frac{1}{\sqrt{2}}\left(|0\rangle + e^{i\theta}|1\rangle\right) \mid \theta = 0, \pi/4, 2\pi/4, \ldots, 7\pi/4\}$ and sends them to Bob. After receiving all the qubits, Bob entangles them according to the universal brickwork state [BFK09]. The computation stage involves interaction: for each layer of the

brickwork state, for each qubit, Alice sends a classical message to Bob to tell him in what basis of the (X, Y) plane he should measure the qubit. Bob performs the measurement and communicates the outcome; Alice's choice of angles in future rounds will depend on these values. Importantly, Alice's quantum states and classical messages are astutely chosen so that, no matter what Bob does, he cannot infer anything about her measurement pattern. If Alice is computing a classical function, the protocol finishes when all the qubits are measured. If she is computing a quantum function, Bob returns to her the final qubits. A modification of the protocol also allows Alice's inputs to be quantum. An *authentication* mechanism is provided; this allows Alice to ascertain that Bob has followed her instructions.

One can view the UBQC protocol as an interactive proof system for any language in BQP where Alice acts as the verifier and Bob as the prover [BFK09, ABE10]. Moreover, the protocol can be adapted to the setting of a purely classical verifier who communicates classically with two noncommunicating entangled provers, in order to perform a blind quantum computation [BFK09]. In this scenario, the general idea is for one prover to be used to prepare the random qubits that would have been generated by Alice in the original protocol, while the other prover is used for universal blind quantum computation. Our main technique is the extension of this view in order to substitute a quantum verifier in a protocol for QMIP with a purely classical one.

4 Contribution

4.1 Definitions

The main step in our construction is to design an interactive protocol with only classical communication that replaces a turn for the verifier in a given quantum interactive proof system: the new protocol requires only classical resources for the verifier. The next definition captures this notion.

Definition 2. *A k-**party delegated quantum computation** is any protocol which accepts quantum input states stored in message registers $\mathsf{M_i}$ in Hilbert space $\mathcal{M}_i$ from each of k provers P_i, and classical input C_V from a verifier V, where C_V represents the classical description of a quantum circuit of size polynomial in $\log\left(\prod_i \dim(\mathcal{M}_i)\right)$. After C_V is applied to $(\mathsf{M_1}, \ldots, \mathsf{M_k})$, the protocol returns to each prover P_i the register $\mathsf{M_i}$ and returns y to V, where y is the classical result of measurements performed on any ancillary qubits introduced by V in C_V. The provers may share prior entanglement, but are not allowed to communicate during the protocol. Furthermore, communication between individual provers and the verifier is purely classical, and the verifier is restricted to performing randomized polynomial-time classical computation.*

In order to provide a full simulation of the quantum verifier, we require that our protocol have certain properties as formalized in the next few definitions. At first glance, some of these properties may seem stronger than what would be required; we expand on this in Section 4.2.

Definition 3. *A k-party delegated quantum computation is* efficient *if:*

- *Given C_V, the description of V can be computed in time polynomial in $O((\prod_i \dim(\mathcal{M}_i))$;*
- *V's computation runs in time polynomial in $O((\prod_i \dim(\mathcal{M}_i))$.*

For a k-party delegated quantum computation, we define the *result of the computation* to be y together with the final state of message registers $(\mathsf{M_1}, \ldots, \mathsf{M_k})$.

Definition 4. *A k-party delegated quantum computation is* authenticated with parameter δ *if there exists a bit $y_0 \in \{\mathsf{pass}, \mathsf{fail}\}$ in y such that:*

- *if all parties follow the protocol, $y_0 = \mathsf{pass}$;*
- *if one or more provers interfere with the protocol then either:*
  - *their actions fail to alter the result of the computation and $y_0 = \mathsf{pass}$; or*
  - *their actions alter the result of the computation, and except with probability at most $2^{-\delta}$, they are detected, with $y_0 = \mathsf{fail}$, indicating alteration.*

In Definition 6, we give a notion of *privacy* in terms of a black-box functionality (defined below) for a multi-party delegated computation. Informally speaking, a private protocol will not leak any information to provers and hence such a protocol can be used for the simulation of a protocol for QMIP.

Definition 5. *A k-**party circuit evaluation black-box** is a functionality which accepts quantum input states from each of k provers P_i stored in message registers $\mathsf{M_i}$ and classical input C_V from a verifier V, where C_V represents the classical description of a quantum circuit. The black-box returns to each prover P_i the register $\mathsf{M_i}$ after applying C_V to $(\mathsf{M_1}, \ldots, \mathsf{M_k})$.*

In the next definition, we use the notion of circuit dimensions to represent the depth and the width of the circuit.

Definition 6. *Let $\mathcal{P}$ be a k-party delegated quantum computation and $\mathcal{B}$ to be the k-party circuit evaluation black-box. For both $\mathcal{P}$ and $\mathcal{B}$, let the verifier's input be any arbitrary circuit C_V and the provers' input be any states stored in $\mathsf{M_1}, \ldots, \mathsf{M_k}$. We say $\mathcal{P}$ is* private *if the distribution of information obtainable by any malicious prover P_i in $\mathcal{P}$ is dependent only on the dimensions of C_V and on the distribution of information obtainable by P_i in $\mathcal{B}$.*

As defined in Section 3, in a quantum interactive proof system, the verifier has a private quantum register V, however such a scenario is not captured in our definition of a multi-party delegated quantum computation. The next lemma removes the requirement of quantum memory for the verifier during the provers' turns. In our construction, we also do away with the private quantum register V during each turn of the verifier: this is explained as part of the proof of Theorem 1.

Lemma 1. *Given any $L \in \mathsf{QMIP}(k, m, c, s)$, there exists an interactive proof system for L where the verifier does not require quantum memory during the provers' turns and only the soundness error changes to $s' \leq \max\{s, \epsilon\}$, for any fixed $\epsilon > 0$.*

Proof. In order to remove the requirement of quantum memory, at the end of her turn, the verifier encrypts and authenticates her quantum register V and sends it to one of the provers and asks him to return them with his message in the next turn. Both encryption and authentication can be achieved by applying a random error correcting code (e.g. [BCG$^+$02]), where the verifier only needs to store a classical key. In this way, only the soundness error might be affected: since the probability of not detecting a cheating player in the authentication protocol is bounded by some ϵ which is exponentially small in the security parameter, we have $s' \leq \max\{s, \epsilon\}$. □

4.2 Main Result

As classical information processing is a special case of quantum information processing, trivially $\mathsf{MIP}^* \subseteq \mathsf{QMIP}$; we prove the reverse inclusion by showing how a multi-party delegated quantum computation can replace the required quantum power of the verifier in a protocol for QMIP with only classical computing.

There are two steps to our main result. First, we show in Theorem 1 that if an efficient, authenticated and private k-party delegated quantum computation exists, then $\mathsf{QMIP} \subseteq \mathsf{MIP}^*$. Then, we show in Protocol 1 how to instantiate such a k-party delegated quantum computation protocol and prove that it satisfies these requirements in Theorem 2. More intuition on Protocol 1 is given after the proof of Theorem 1.

As mentioned, we have imposed some conditions on the k-party delegated quantum computation protocol in Theorem 1 that are stronger than may seem necessary. We now elaborate on this. First of all, in a given protocol for QMIP, the verifier's circuits are public. Thus, our privacy definition would seem to be stronger than what is necessary as only the verifier's private register V needs to be hidden (we also must ensure that the protocol does not provide a covert means of communication between provers). However, for our instantiation of the k-party delegated quantum computation protocol, the authentication property heavily relies on this strong definition of privacy; so does the construction that sees P_1 manipulate the encrypted version of the verifier's private register V and of

the other provers' message registers. Secondly, the authentication definition seems stronger than required: all we would really need to guarantee is that the result that is returned to the verifier, y, be correct. However, also due to the construction that sees P_1 manipulate the encrypted and authenticated version of the verifier's private register V, we must have that not just y, but the register in P_1's hands be authenticated at each iteration of the k-party delegated quantum computation protocol.

Theorem 1. *If there exists a k-party delegated quantum computation protocol which is efficient, authenticated (with parameter δ) and private then $\mathsf{QMIP}(k, m, c, s) \subseteq \mathsf{MIP}^*(k, m', c, s')$, where $s' \leq \max\{s, 2^{-\delta}\}$ and the number of turns m' is polynomially bounded in the input size $|x|$.*

Proof. Recall that by Lemma 1, it is sufficient to consider a verifier with no quantum memory during the provers' turns (the only consequence being a potential modification of the soundness error; this is addressed below).

Without loss of generality, we assume that the interaction begins with a turn of the provers. Hence the interactive proof can be broken down into a number of *rounds* polynomial in the input size, where in each round j:

1. Each prover applies a quantum circuit to prepare the new state of message register $\mathsf{M_i}$.
2. Each prover P_i transmits his message register $\mathsf{M_i}$ to the verifier.
3. The verifier performs a computation described by circuit V^j on $(\mathsf{V}, \mathsf{M_1}, \ldots, \mathsf{M_k})$.
4. The verifier transmits the message register $\mathsf{M_i}$ to each prover P_i.

We wish to use $\mathcal{P}$, a k-party delegated quantum computation protocol, to perform the computation and message preparation required by steps 2–4. However, $\mathcal{P}$ requires a fully classical verifier. Even with the application of Lemma 1, we must still deal with the verifier's quantum register V in Step 3. The solution is for the register V to always remain with P_1 in an encrypted and authenticated form: V adjusts her circuits to add the encryption and authentication at the end of the circuit V^j, leaving the encrypted and authenticated register V with P_1, and then circuit V^{j+1} is adapted so that it acts on the encrypted and authenticated register.

Using the above construction, we may now use $\mathcal{P}$ to perform the computation and message preparation required by steps 2–4 above, yielding a verifier performing only classical randomized polynomial-time computation, and hence a protocol in MIP^* (we specify that if $y_0 = \mathsf{fail}$ for any call to $\mathcal{P}$, the verifier rejects). It remains to show that the parameters are as indicated in the statement of Theorem 1.

Clearly, the number k of provers is unchanged. Note that $\mathcal{P}$ is efficient, since the verifier's strategy can be computed in polynomial-time, runs in polynomial time, and requires only polynomial communication. Hence the resulting protocol still has a polynomial number of turns, leading to m' being polynomially bounded in the input size $|x|$.

To see that the completeness parameter, c, is unchanged, note that if $x \in L$, and the honest provers follow the protocol (including calls to $\mathcal{P}$), the verifier will accept with probability c as calls to $\mathcal{P}$ always produce the correct state of the message registers.

Finally, the soundness error, s, is affected by the following:

1. By Lemma 1, (and setting $\epsilon = 2^{-\delta}$), the soundness error for the protocol with a memoryless verifier is $\leq \max\{s, 2^{-\delta}\}$. This is consistent also with the use of $\epsilon = 2^{-\delta}$ as a security parameter for the authentication of V.
2. Since $\mathcal{P}$ is private, the distribution of information gained from $\mathcal{P}$ by any cheating prover P_i is no more than that obtained through a black-box functionality and circuit width and depth of the verifier's computation. This is no more than what is obtainable in the original interactive proof, as in an interactive proof the verifier's circuit dimensions are known. Furthermore, note that our black-box definition of privacy precludes any leaking of information (other than circuit dimensions) during sequential composition, and calls to $\mathcal{P}$ are independent. Hence cheating provers cannot exploit calls to $\mathcal{P}$ to increase the soundness error.
3. Since P is authenticated, any deviation from the computation V^j leads to the verifier rejecting (since $y_0 = \mathsf{fail}$), except with probability exponentially small in the security parameter δ.

The soundness error of the final protocol is independently affected by 1–3 above; this leads to the soundness error for the final protocol satisfying $s' \leq \max\{s, 2^{-\delta}\}$. □

We now explicitly formulate a k-party delegated quantum computation and show that the protocol satisfies the conditions of efficiency, authentication and privacy required by Theorem 1. As outlined in Section 2, the idea behind our construction, given in Protocol 1, is to use the blind quantum computation protocol [BFK09] between the verifier and entangled provers to remove the need for the verifier to exchange and process quantum information.

In order to run a quantum circuit, the verifier enlists the help of two entangled provers, P_1 and P_2. P_1 first receives authenticated and encrypted message registers $\mathsf{M_i}$ teleported from the other provers (the special case of authenticating P_1's own message register is also covered), and then the two-server blind quantum computation protocol is executed with P_1 and P_2: P_2 helps prepare the required initial qubits for the UBQC protocol by performing measurements on his share of the entangled states, while P_1 performs the actual blind computation, eventually computing the final states of the encrypted message registers. These registers are re-distributed (via teleportation) to all other provers, who then verify their authenticity with the help of the verifier.

The input message preparation protocol (Protocol 2) accomplishes two things: it transfers all of the provers's quantum message registers to P_1, and also encrypts and authenticates them so that no prover can extract any information, any tampering will be detected and the authentication is compatible with the UBQC protocol. The authentication consists of the verifier instructing the provers to encode their message qubits into an error correcting code, telling them which *trap* qubits to insert (these are qubits that are in a known eigenstate of a Pauli operator), and how to permute the resulting states. The provers perform a teleportation measurement involving their system and the shared entanglement with P_1, but instead of revealing the measurement result to P_1, they send it to the verifier: this action transfers their encrypted message register to P_1. There is an extra step for Prover P_2 since he is involved later on in the preparation of the initial states for the blind protocol: the position of trap qubits inserted by prover P_2 in his message register should become hidden to him even after they are teleported to prover P_1, this is done in Step 4 in Protocol 2. A similar issue is valid for P_1's own message register: Step 5 in Protocol 2 involves P_2 who applies an additional permutation to P_1's message register so that the position of the trap qubits remains hidden to both P_1 and P_2. On top of this, all qubits in message registers receive random Z-rotations; these are necessary in order for the inputs to be compatible with the UBQC protocol (Protocol 4).

At this point, the verifier, together with P_1 and P_2, execute the UBQC with entangled servers protocol (Protocol 4 in the Appendix), where the encoding and authentication is already performed, and where P_1 provides a quantum input. More explicitly, Protocol 4 is an adaptation of the authenticated UBQC protocol with quantum input and output to the entangled servers scenario. The full protocol is a simple consequence of our previous result [BFK09].

The output message distribution protocol, Protocol 3, consists of sending the appropriate encoded quantum message register computed by prover P_1 to provers P_i for verification and decryption. The authentication in our protocol ensures that both P_1 and P_2 are performing the required operations; it also ensures that P_1 has sent the correct message registers to each P_i. In order to do so, the verifier instructs each prover to make several Pauli measurements over specific qubits. Note that due to blindness of Protocol 4 and initial random permutation, the provers have no information about the location and the state of the returned trap qubits. Furthermore, the returned qubits from P_1 are all one-time padded since the result of the teleportation measurements are known only to the verifier. Hence Protocol 3 reveals at most the new location of trap qubits of each prover P_i which is of course independent of the state of the message registers and hence the obtained protocol is authenticated and private. This is formalized in Theorem 2. The blindness property of Protocol 4 guarantees that no information is leaked to the provers, as if V herself is running the quantum circuits. The overhead cost of such action is the usage of a number of copies of $|\Phi^+\rangle$ linear in the size of the verifier's quantum circuits and polynomially many classical messages.

Protocol 1 k-Prover Delegated Quantum Computation Protocol

1. Using Protocol 2, the input message registers of all provers are encoded, encrypted and transferred to prover P_1.
2. The verifier uses P_1 and P_2 and executes the Authenticated UBQC with Entangled Provers with Quantum Input and Output Protocol (Protocol 4 in the Appendix) with the encoded message registers received by P_1 in Step 1 (with keys k^X and k^Z as defined in Protocol 2).
3. Using Protocol 3, the message registers resulting from the computation in Step 2 are distributed and verified.

Protocol 2 Input Message Preparation Protocol

1. The verifier chooses $\mathbb{C}$, where $\mathbb{C}$ is some $n_{\mathbb{C}}$-qubit error-correcting code with distance $d_{\mathbb{C}}$. The security parameter is $\delta = d_{\mathbb{C}}$.
2. For each prover P_i, for each qubit j in message register $\mathsf{M_i}$:
   (a) The verifier instructs P_i to encode qubit j using $\mathbb{C}$.
   (b) The verifier instructs P_i to prepare $3n_{\mathbb{C}}$ qubits in eigenstates of Pauli operators chosen uniformly at random by the verifier. We refer to these as *trap* qubits and we refer to the trap qubits and the qubits used in the error-correcting code collectively as a *block*.
   (c) The verifier instructs P_i to apply a random permutation $\pi_{i,j}$ to the block.
   (d) The verifier instructs each P_i except P_1 to apply to each qubit k a random Z-rotation, $Z(\theta_k)$, with $\theta_k \in \{0, \pi/4, 2\pi/4, \ldots, 7\pi/4\}$, independently chosen for each qubit.
3. Each prover P_i except P_1 uses the teleportation protocol to transmit their entire quantum message register to P_1 but communicates the measurement results only to the verifier.
4. For each block j, the verifier instructs P_1 to apply a permutation $\pi'_{2,j}$ to the qubits containing the result of the teleportation from P_2.
5. For each block j of prover P_1:
   (a) P_1 uses the teleportation protocol to transmit his $4n_{\mathbb{C}}$ qubits to P_2 but communicates the measurement results only to the verifier.
   (b) The verifier instructs P_2 to apply a permutation $\pi'_{1,j}$ to the qubits containing the result of the teleportation from P_1. The verifier also instructs P_2 to apply to each qubit k of the system a random Z-rotation in $Z(\theta_k), \theta_k \in \{0, \pi/4, 2\pi/4, \ldots, 7\pi/4\}$, independently chosen for each qubit.
   (c) P_2 uses the teleportation protocol to return the $4n_{\mathbb{C}}$ qubits to P_1 but communicates the measurement results only to the verifier.

Note that for each qubit, the verifier can now take into account all the received teleportation results as well as the $Z(\theta_k)$ rotations, to compute the X operation and Z-rotation that has been applied to the original message registers. Let these values be $k_j^X \in \{0, 1\}$ and $k_j^Z \in \{0, \pi/4, 2\pi/4, \ldots, 7\pi/4\}$.

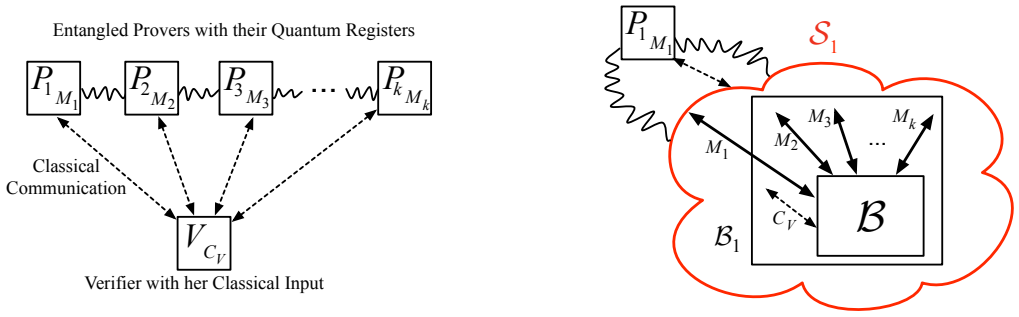
Theorem 2. *Protocol 1 is a k-party delegated quantum computation which is efficient, authenticated (with parameter δ) and private for all $k \geq 2$.*

Proof. If all parties follow Protocol 1, the outcome is correct; this follows from the correctness of the UBQC protocol. In particular, Protocols 2 and 3 serve only to encode and decode the input states for use in blind quantum computation (Protocol 4 in the Appendix); no information is lost during either of these stages, and so the correctness relies only on the correctness of Protocol 4 (see the Appendix). Additionally, Protocol 1 is efficient since our construction is polynomial-time computable and the verifier runs in time polynomial in $O\left(\prod_i \dim(\mathcal{M}_i)\right)$.

To show that Protocol 1 is private, we construct a set of *simulators* $\{\mathcal{S}_1, \ldots, \mathcal{S}_k\}$ (see Figure 1). Each simulator $\mathcal{S}_i$ interacts with P_i, simulating all other participants. As $\mathcal{S}_i$ simulates all participants except P_i, this implies that $\mathcal{S}_i$ shares entanglement with P_i (as P_j, $j \neq i$ would) and

Protocol 3 Output Message Distribution and Verification Protocol

1. For each prover $P_i \neq P_1$, the verifier instructs P_1 to apply a permutation $\pi''_{i,j}$ to each block j and then teleport the $4n_i n_{\mathbb{C}}$ qubits of message register $\mathsf{M_i}$ to P_i. The measurement results are communicated only to the verifier.
2. The verifier instructs P_i to measure each of the trap qubits in their correct basis.
3. P_i returns the measurement results of the trap qubits to the verifier.
4. If the verifier receives the expected measurement results for each of the trap qubits, she sets $y_0 = \mathsf{pass}$. If a mismatch is found, she sets $y_0 = \mathsf{fail}$.
5. If $y_0 = \mathsf{pass}$, for each P_i:
   (a) The verifier sends P_i the one-time pad key (coming from Step 1 above and from the UBQC with entangled servers protocol), and all permutations applied to their qubits.
   (b) P_i decrypts and decodes their message register.



(a) Real Protocol

(b) Simulation Protocol

Figure 1: (a) A k-party delegated quantum computation protocol, with the provers and their quantum registers $\mathsf{M_1}, \ldots, \mathsf{M_k}$ and the verifier and her classical input C_V. (b) A simulation protocol where simulator S_1 simulates all other participants $P_2, \ldots, P_k$ while sharing entanglement with P_1 and only communicating classically with P_1. The simulator also has access to a black box $\mathcal{B}_1$ with input register M_1, that calls the k-party circuit evaluation black-box $\mathcal{B}$ with fixed inputs in registers $\mathsf{M_j}$ for all $j \neq 1$ and C_V.

communicates classically with P_i (as V would). Importantly, $\mathcal{S}_i$ does not have access to any of the other participant's inputs, but instead has access to a single use of a black box $\mathcal{B}_i$ that takes as input a single quantum register M_i, and calls the k-party circuit evaluation black-box $\mathcal{B}$ with fixed inputs in registers M_j for all $j \neq i$, corresponding to the contents of registers M_j in the protocol, as well as the classical description of C_V corresponding to V's input. $\mathcal{S}_i$ also has access to the circuit dimensions of C_V. $\mathcal{B}_i$ calls the k-party circuit evaluation black-box $\mathcal{B}$, outputing only register M_i to $\mathcal{S}_i$ (again, see Figure 1). We show that no P_i can distinguish between an execution of the real protocol and an execution of the protocol with $\mathcal{S}_i$. It follows that the protocol is private since this proves that the distribution of information obtainable in Protocol 1 is dependent only on the circuit dimensions of C_V and on the distribution of information obtainable by P_i in $\mathcal{B}$ (since otherwise $\mathcal{S}_i$ would not be indistinguishable). Note that we consider here *perfect* indistinguishability for a computationally unbounded P_i and that we make no restrictions on P_i's *a priori* knowledge.

In what follows, we assume that $\mathcal{S}_i$ and P_i share the same entangled states as P_i would share with other provers P_j $(j \neq i)$ in a real execution of the protocol. Recall that in the protocol, all communication is classical and is between P_i and V, hence this is the only communication that $\mathcal{S}_i$ must simulate.

First, we consider the case for $i \geq 3$ and give a description for $\mathcal{S}_i$. In Protocol 1, the involvement of P_i is limited, as P_i simply teleports his authenticated and encrypted input registers to P_1 (Protocol 2) and later receive his answer register encoded in a similar fashion (Protocol 3). Thus the strategy for $\mathcal{S}_i$ is, following the same format as in Step 2 of Protocol 2, to instruct P_i to encode and encrypt his quantum input register according to a random key chosen by $\mathcal{S}_i$, to receive (Step 3 of Protocol 2) the state by teleportation via the entanglement that P_i would normally share with P_1, but that $\mathcal{S}_i$ is actually sharing with P_i. $\mathcal{S}_i$ then decodes the state (recall that $\mathcal{S}_i$ knows the key as well as the results of the teleportation measurements). This state is used as the input into the black-box $\mathcal{B}_i$. The output is then re-encoded in a manner consistent with what is used in Protocol 3 and teleported back to P_i for verification. Clearly, from the point of view of P_i, his interaction with $\mathcal{S}_i$ is indistinguishable from his interaction with the real protocol.

A similar strategy works for $\mathcal{S}_2$: the input and output registers are dealt with as above, with P_2's input register used as input in the black-box $\mathcal{B}_2$. Again, the output is also re-encoded and returned to P_2 as above. Additionally, in Step 5 of Protocol 2, $\mathcal{S}_2$ asks P_2 to apply a random permutation to a system that remains encrypted from P_2's point of view. This is indistinguishable from P_1's real input due to the quantum one-time pad applied by the teleportation, and thus S_2 need only provide P_2 with the completely mixed state. $\mathcal{S}_2$ asks P_2 to apply random Z-rotations, as in Step 5 of Protocol 2. Furthermore, P_2 is involved in Step 2 of Protocol 1, which calls Protocol 4 in the Appendix. For Protocol 4, since $\mathcal{S}_2$ knows the dimensions of the circuit, he can choose the dimensions n and m of the brickwork state accordingly. As for P_2's involvement in Step 2 of Protocol 4, he is asked to measure his part of $\left|\Phi^+_{x,y}\right\rangle$ in $|\pm_{\tilde{\theta}_{x,y}}\rangle$ as usual, the difference being that the measurement results $m_{x,y}$ are not used any further. From the point of view of P_2, his interaction with $\mathcal{S}_2$ is indistinguishable from his interaction with the real protocol.

To show a strategy for $\mathcal{S}_1$, we analyze each step of Protocol 1 separately:

- **Step 1.** This is a call to Protocol 2. The strategy for $\mathcal{S}_1$ is to instruct P_1 to encode and encrypt his quantum input register according to a random key chosen by $\mathcal{S}_1$. $\mathcal{S}_1$ then receives (Step 5 of Protocol 2) the state by teleportation via the entanglement that P_1 would normally share with P_2, but that $\mathcal{S}_1$ is actually sharing with P_1. $\mathcal{S}_1$ then decodes the state (recall that $\mathcal{S}_1$ knows the key as well as the teleportation bits). This state is used as the input into the black-box $\mathcal{B}_1$.
- **Step 2.** Here, Protocol 4 is executed. As above, since $\mathcal{S}_1$ knows the dimensions of the circuit, he can choose the dimensions n and m of the brickwork state accordingly. Due to the blindness of the UBQC protocol (Protocol 4 in the Appendix), the interaction involved can be simulated knowing only the circuit dimensions: $\mathcal{S}_1$'s strategy for this step is simply to instruct P_1 to measure the brickwork state qubits with random measurement angles. Using this technique, P_1 cannot distinguish between his interaction with the real protocol or with $\mathcal{S}_1$.
- **Step 3.** The output from the black-box $\mathcal{B}_i$ is re-encoded (in a manner consistent with what is used in Protocol 3). S_1 then applies a Hadamard gate and subsequent Z rotation to each qubit. The angle of rotation for each qubit is chosen independently as a random multiple of $\frac{\pi}{4}$. This system is then teleported to the correct position in the column $x = n-1$ of the brickwork state via the entanglement established in Step 1 of Protocol 4. $\mathcal{S}_1$ uses the knowledge of P_1's measurement angles and results in order to use the last layer of the brickwork state to undo the random Z rotation and Hadamard on each qubit and to establish the one-time pad key on the system.

  The verification and decryption procedure is executed according to Protocol 3.

From the point of view of P_1, his interaction with $\mathcal{S}_1$ is indistinguishable from his interaction with the real protocol. Thus, as the required set of simulators $\{\mathcal{S}_1, \ldots, \mathcal{S}_k\}$ exists, Protocol 1 is private.

Finally, to show that Protocol 1 is authenticated, first note that if all parties follow the protocol, the outcome is correct as discussed above and the verifier sets $y_0 = \mathsf{pass}$ in Protocol 3 since all trap qubit verifications succeed. We now show that any non-trivial modification of the result of the computation leads to $y_0 = \mathsf{fail}$, except with exponentially small probability in δ. As no prover has any information about the layout of the trap qubits (due to random permutation performed in Protocol 2), the probability of disturbing a trap qubit by Protocol 1 is the same as in Protocol 4. Thus any interference with the delegated quantum computing protocol by any group of provers can be

detected due to the authentication property of Protocol 4 (this is based on Theorem 7 of [BFK09] see also the Appendix). That is to say, in order to create an undetected error, a malicious party must apply an operator of weight at least δ, but as they do not know the location of the trap qubits, the probability of applying such an operator without disturbing a trap qubit scales as $2^{-\delta}$.

Moreover, it is straightforward to verify that from the point of view of the verifier, any deviation by any number of provers in the Input Message Preparation Protocol (Protocol 2) and/or the Output Message Distribution and Verification Protocol (Protocol 3) is equivalent to a deviation performed by Prover 1 during the course of Step 2 of Protocol 1, or more precisely, during Step 5 of Protocol 4 in the Appendix. For example if the provers attempt to collectively cheat by not performing the required permutations, the trap qubits will be misplaced from the point of view of the verifier, and since in the Output Message Distribution Protocol (Protocol 3) all the message registers are one-time padded, no prover can correctly produce the expected measurement result over the trap qubits. Therefore the verification steps will fail except in the case where the provers can correctly guess the expected measurement results or the permutations applied, which occurs only with exponentially small probability. Any other type of deviation, such as not performing the correct encoding of the message register, sending the wrong measurement result during the teleportation procedure, or incorrect preparation of the trap qubits is detected for exactly the same reason due to the authentication property of Protocol 4. □

5 Discussion

In summary we proved $\mathsf{QMIP}[k] = \mathsf{MIP}^*[k]$ ($k \geq 2$) based on the existence of a protocol for k-party delegated quantum computation which is efficient, authenticated and private. Combined with the results of [JJUW09], we get that $\mathsf{QMIP}[k] = \mathsf{MIP}^*[k]$ for all k. Our proof is based on novel techniques that give a direct simulation for a quantum interactive proof system with a classical interactive proof system with entangled provers. These techniques may have interesting applications elsewhere, but do not appear to be directly applicable to the single prover setting; thus the relationship between our approach and that of [JJUW09] remains an interesting question.

We have showed how to use the power of shared entanglement between provers to replace the quantum communication in a protocol for QMIP. While the longstanding open problem regarding the relationship between QMIP and MIP is still open, our result demonstrates that it suffices to look at the direct relation between MIP^* and MIP, forgetting QMIP altogether, and focusing the question to the understanding of the power of shared entanglement.

Acknowledgements

We are grateful to Gus Gutoski for his insightful input. We would also like to thank Scott Aaronson, Iordanis Kerenidis and Sarvagya Upadhyay for helpful comments and discussions. We are grateful for the hospitality of the Bellairs Research Institute of McGill University where an important portion of this work was accomplished. This work is partially supported by Singapore's National Research Foundation and Ministry of Education and by the British Engineering and Physical Sciences Research Council (EP/E059600/1).

References

[ABE10] D. Aharonov, M. Ben-Or, and E. Eban. Interactive proofs for quantum computations. In *Proceedings of Innovations in Computer Science (ICS 2010)*, pages 453–469, 2010.

[AKN98] D. Aharonov, A. Kitaev, and N. Nisan. Quantum circuits with mixed states. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC '98)*, pages 20–30, 1998.

[ALM+98] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45:501–555, 1998.

[AS98] S. Arora and S. Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45:70–122, 1998.

[Bab85] L. Babai. Trading group theory for randomness. In *Proceedings of the 7th Annual ACM Symposium on Theory of Computing (STOC '85)*, pages 421–429, 1985.

[BBC+93] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, and W. K. Wootters. Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels. *Physical Review Letters*, 70:1895–1899, 1993.

[BCG+02] H. Barnum, C. Crépeau, D. Gottesman, A. Smith, and A. Tapp. Authentication of quantum messages. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science (FOCS '02)*, pages 449–458, 2002.

[BFK09] A. Broadbent, J. Fitzsimons, and E. Kashefi. Universal blind quantum computation. In *Proceedings of the 50th Annual Symposium on Foundations of Computer Science (FOCS 2009)*, pages 517–527, 2009.

[BFL91] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991.

[BOGKW88] M. Ben-Or, S. Goldwasser, J. Kilian, and A. Wigderson. Multi-prover interactive proofs: how to remove intractability assumptions. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC '88)*, pages 113–131, 1988.

[CHTW04] R. Cleve, P. Hoyer, B. Toner, and J. Watrous. Consequences and limits of nonlocal strategies. In *Proceedings of the 19th Annual IEEE Conference on Computational Complexity (CCC '04)*, pages 236–249, 2004.

[DK06] V. Danos and E. Kashefi. Determinism in the one-way model. *Physical Review A*, 74:052310 [6 pages], 2006.

[DKP07] V. Danos, E. Kashefi, and P. Panangaden. The measurement calculus. *Journal of the ACM*, 54:8 [45 pages], 2007.

[FGL+96] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43:268–292, 1996.

[FRS94] L. Fortnow, J. Rompel, and M. Sipser. On the power of multi-prover interactive protocols. *Theoretical Computer Science*, 134:545–557, 1994.

[GMR89] S. Goldwasser, S. Micali, and C. W. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18:186–208, 1989.

[JJUW09] R. Jain, Z. Ji, S. Upadhyay, and J. Watrous. QIP=PSPACE. To appear in *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC 2010)*. Available as `arXiv:0907.4737[quant-ph]`, 2009.

[JUW09] R. Jain, S. Upadhyay, and J. Watrous. Two-message quantum interactive proofs are in PSPACE. In *Proceedings of the 50th Annual Symposium on Foundations of Computer Science (FOCS 2009)*, pages 534–544, 2009.

[KKMV08] J. Kempe, H. Kobayashi, K. Matsumoto, and T. Vidick. Using entanglement in quantum multi-prover interactive proofs. In *Proceedings of the 23rd IEEE Annual Conference on Computational Complexity (CCC '08)*, pages 211–222, 2008.

[KM03] H. Kobayashi and K. Matsumoto. Quantum multi-prover interactive proof systems with limited prior entanglement. *Journal of Computer and System Sciences*, 66:429–450, 2003.

[KV06] J. Kempe and T. Vidick. On the power of entangled quantum provers. Available as `arXiv:quant-ph/0612063`, 2006.

[KW00] A. Kitaev and J. Watrous. Parallelization, amplification, and exponential time simulation of quantum interactive proof systems. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC 2000)*, pages 608–617, 2000.

[LFKN90] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proofs. In *Proceedings of the 31st Annual IEEE Symposium on Foundations of Computer Science (FOCS '90)*, pages 1–10, 1990.

[RB01] R. Raussendorf and H. J. Briegel. A one-way quantum computer. *Physical Review Letters*, 86:5188 – 5191, 2001.

[RBB03] R. Raussendorf, D. E. Browne, and H. J. Briegel. Measurement-based quantum computation with cluster states. *Physical Review A*, 68:022312 [32 pages], 2003.

[Sha90] A. Shamir. IP=PSPACE. In *Proceedings of the 31st Annual IEEE Symposium on Foundations of Computer Science (FOCS '90)*, pages 11–15, 1990.

[Wat99] J. Watrous. PSPACE has constant-round quantum interactive proof systems. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science (FOCS '99)*, pages 112–119, 1999.

A Authenticated UBQC with Entangled Servers with Quantum Input and Output

In this section, we explicitly construct the full protocol for authenticated UBQC with entangled servers with quantum input and output received and stored by Prover 1. This is a simple composition and adaptation of several protocols in [BFK09] with an extra construction (Step 3g) to deal with the the fact that the verifier's circuit is now public knowledge. In the Protocol 4, we assume that at step 4, the verifier's transformed circuit is implemented with a pattern on a brickwork state $\mathcal{G}_{n\times m}$ with measurements given as multiples of $\pi/4$. Each qubit $|\psi_{x,y}\rangle \in \mathcal{G}_{n\times m}$ is indexed by a column $x \in \{0,\ldots,n\}$ (column 0 consists of the input qubits and column n of the output qubits) and a row $y \in \{1,\ldots,m\}$. Thus each qubit is assigned: a measurement angle $\phi_{x,y}$, a set of X-dependencies $D_{x,y} \subseteq [x-1]\times[m]$, and a set of Z-dependencies $D'_{x,y} \subseteq [x-1]\times[m]$. Here, we assume that the dependency sets $X_{x,y}$ and $Z_{x,y}$ are obtained via the flow construction [DK06]. During the execution of the pattern, the actual measurement angle $\phi'_{x,y}$ is a modification of $\phi_{x,y}$ that depends on previous measurement outcomes in the following way: let $s^X_{x,y} = \oplus_{i\in D_{x,y}} s_i$ be the parity of all measurement outcomes for qubits in $X_{x,y}$ and similarly, $s^Z_{x,y} = \oplus_{i\in D'_{x,y}} s_i$ be the parity of all measurement outcomes for qubits in $Z_{x,y}$. Then $\phi'_{x,y} = (-1)^{s^X_{x,y}}\phi_{x,y} + s^Z_{x,y}\pi$.

Protocol 4 is a slight modification of the UBQC with entangled servers protocol which was shown to be correct in Section 5 based on Theorem 2 in [BFK09]. The only difference is in Step 3g, where several identity gates are added which does not affect the correctness. A simple modification in Protocol 4 ensures that the inputs are coming from the provers, without affecting the blindness property (Theorem 6 in [BFK09]). Finally the extra padding structure added in Step 3g allow us to conveniently extend the proof of authentication (Theorem 7 in [BFK09]) for the case of the public knowledge of verifier's circuit available in a QMIP protocol.

Protocol 4 Authenticated UBQC with Entangled Provers and Quantum Input and Output

1. **Initial input of the protocol**
   (a) Provers 1 and 2 share $\left|\Phi^+_{x,y}\right\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ for $x = 1, \ldots, n-1$ and $y = 1, \ldots, m$ (n and m are given in step 3g below).
   (b) Prover 1 receives quantum input prepared following the Step 1 of Protocol 1 to be the first column of the brickwork state in the computation. These input qubits are encoded with $\mathbb{C}$ ($\mathbb{C}$ is some $n_{\mathbb{C}}$-qubit error-correcting code with distance $d_{\mathbb{C}}$) and $3n_{\mathbb{C}}$ qubits in eigenstates of Pauli operators chosen uniformly at random (called trap qubits) are added to them. The collection of these $4n_{\mathbb{C}}$ qubits are called a *block*. Moreover a random permutation is applied to each block and finally a one-time pad with keys $k^Z_{0,y} \in_R \{0, \pi/4, 2\pi/4, \ldots, 7\pi/4\}$ and $k^X_{0,y} \in_R \{0, 1\}$ is performed on each qubit.
   (c) The permutations, the state of trap qubits and one-time pad keys are known to the verifier and not to provers 1 and 2.
2. **Verifier's preparation with Prover 2**
   For each column $x = 1, \ldots, n-1$ and for each row $y = 1, \ldots, m$:
   (a) The verifier chooses $\tilde{\theta}_{x,y} \in_R \{0, \pi/4, 2\pi/4, \ldots, 7\pi/4\}$ and sends it to Prover 2, who measures his part of $\left|\Phi^+_{x,y}\right\rangle$ in $|\pm_{\tilde{\theta}_{x,y}}\rangle$.
   (b) Prover 2 sends $m_{x,y}$, the outcome of his measurement, to the verifier.
3. **Verifier's authenticated preparation**
   (a) Convert target circuit C_V to fault-tolerant circuit:
   – Use error-correcting code $\mathbb{C}$.
   – Perform all gates and measurements fault-tolerantly.
   – Some computational basis measurements are required for the fault-tolerant implementation (for verification of ancillae and non-transversal gates). Each measurement is accomplished by making and measuring a *pseudo-copy* of the target qubit: a CTRL-X is performed from the target to an ancilla qubit initially set to $|0\rangle$, which is then measured in the Z-basis.
   – Ancilla qubit wire are evenly spaced through the circuit.
   – The ancillae are re-used. All ancillae are measured at the same time, at regular intervals, after each fault-tolerant gate (some outputs may be meaningless).
   (b) Within each encoded qubit, permute all wires consistent with the position of non-trap qubits in the encoded input of Prover 1, keeping these permutations secret from Provers 1 and 2.
   (c) Within each encoded qubit, add $3n_T$ trap wires consistent with the position of trap qubits in the encoded input of Prover 1. The trap qubit wire (at this point) does not interact with the rest of the circuit.
   (d) Trap qubits are verified using the same ancillae as above: they are rotated into the computational basis, measured using the pseudo-copy technique above, and then returned to their initial basis.
   (e) Any fault-tolerant measurement is randomly interspersed with verification of $3n_T$ random trap wires. For this, identity gates are added as required.
   (f) For encoded qubits with classical outputs, the trap wires of the corresponding blocks are rotated as a last step, so that the following measurement in the computational basis is used for a final verification.
   (g) Convert the whole circuit above to a measurement-based computation on the brickwork state, with the addition of regular Z-basis measurements corresponding to the measurements on ancillae qubits above. Swap and identity gates are added as required, and trap qubits are left untouched. Further identity gates are added so that for any choice of permutation for each block, the resulting dimensions, n and m, of the brickwork states are identical and hence dependent only on the dimensions of C_V.

Protocol 4 — Continued

4. **Verifier's blind quantum computation with Prover 1**
   Taking $\theta_{x,y} = \tilde{\theta}_{x,y} + m_{x,y}\pi$ (for $x = 1, \ldots, n-1$ and $y = 1, \ldots, m$) and $\theta_{0,y} = k^Z_{0,y}$ (for $y = 1, \ldots, m$), the verifier runs the following steps to implement the authenticated measurement pattern constructed in Step 3.
   (a) The verifier periodically instructs Prover 1 to measure in Z as indicated in Step 3. These qubits are chosen at regular spacial intervals so that no information about the structure of the computation is revealed.
   (b) Prover 1 prepares the last column of qubits $|\psi_{n,y}\rangle = |+\rangle$ $(y = 1, \ldots, m)$.
   (c) Prover 1 creates an entangled state from all received qubits, according to their indices, by applying CTRL-Z gates between the qubits in order to create a brickwork state $\mathcal{G}_{n\times m}$.
   (d) For each column $x = 0, \ldots, n-1$ and for each row $y = 1, \ldots, m$:
      i. Verifier computes $\phi'_{x,y}$ with the special case $\phi'_{0,y} = (-1)^{k^X_{0,y}}\phi_{0,y}$.
      ii. The verifier chooses $r_{x,y} \in_R \{0, 1\}$ and computes $\delta_{x,y} = \phi'_{x,y} + \theta_{x,y} + \pi r_{x,y}$.
      iii. The verifier transmits $\delta_{x,y}$ to Prover 1.
      iv. Prover 1 measures in the basis $\{|+_{\delta_{x,y}}\rangle, |-_{\delta_{x,y}}\rangle\}$.
      v. Prover 1 transmits the result $s_{x,y} \in \{0, 1\}$ to the verifier.
      vi. If $r_{x,y} = 1$ above, the verifier flips $s_{x,y}$; otherwise she does nothing.
   (e) The last column consists of quantum outputs together with their corresponding trap qubits, where all these qubits are one-time padded with keys known only to the verifier (Theorem 4 [BFK09]). In order to obtain a classical outcome, the verifier instructs Prover 1 to measure the corresponding block (a quantum output together with its trap qubits). Note that the output measurements are restricted to be in the (X, Y) plane so that the classical outputs will be one-time padded with keys $r_{n-1,y}$ known only to the verifier. However this is not a restriction as any other arbitrary measurement can be also applied by adding the required rotation as part of the original pattern computation. Finally also note that since trap qubits are inserted in each block and each block is randomly permuted, only the verifier knows the exact position of the output qubits.

5. **Verifier's verification**
   The verifier uses the results of the trap qubit measurements performed by Prover 1 above to detect any deviation from the protocol. The verifier accepts only if all the results correspond with her initial preparation of the trap qubits, otherwise she rejects.